**Aufgabe 2**

Im Feld „Benutzername“ und „Email“ wurde die Eingabe einfach in als Variable gesichert und in die Textdatei geschrieben. Soweit endstanden keine Probleme. Bei der HTML Ausgabe der Newslettereinträge jedoch, haben Benutzereingaben wie z.B. <script>alert(?XSS?);</script> zu ungewollten Script-Aufrufen geführt. Um dies zu unterbinden haben wir die Eingaben durch htmlspecialchars() in HTML Code umgewandelt und mit strip\_tags() die Tags entfernt.

**Aufgabe 3 – Code Review**

*Allgemeine Anmerkungen:*

* Terminfindung hat gut funktioniert
* Review Termin ist gut verlaufen, beide Teams haben respektvoll die Verbesserungsvorschläge aufgelistet
* Das Code Review wurde von beiden Teams für hilfreich und sinnvoll befunden
* Verbesserungsvorschläge wurden bereits in die Tat umgesetzt (z.B.: Angepasste Bedienelemente, Textgröße der Allergene)

*Unser Review des andren Teams:*

• Sind Dateien, Variablen, Funktionen, usw. zweckdienlich benannt?

* Ja

• Existieren Fehler im Quelltext?

* Ausgegebene Fehlermeldungen werden nach komplettem schließen und neuem Aufruf der Seite, immer noch angezeigt.
* Bei Eingabe eines Leerzeichens als Name im Newsletter

funktioniert die Seite nicht mehr

• Was würden Sie anders strukturieren?

* CSS styling sollte in eigene Datei ausgelagert werden um die Übersichtlichkeit zu erhalten.
* CSS-Quelltext ohne Struktur.
* $link=mysli\_connect wurde in Zeile 31,63 und 100 initialisieret. Einmal außerhalb der Funktionen hätte gereicht.

Außerdem muss man dann auch nur einmal checken ob der Verbindungsaufbau erfolgreich war.

• Was fällt Ihnen sonst am Quelltext auf?

* Zeile 51-54 hätte man in der sql query mit dem Befehl LIMIT 5 erledigen können.
* Zeile 57 und somit auch die Funktion sort\_meals\_by\_name hätte man in der sql query mit dem Befehl ORDER BY name ASC erledigen können.
* Die Funktion „extractAllergenCodes(&$meals)“ kann man mit einem Left Joins der Tabelle gericht\_hat\_allergen ersetzen.

• Was haben Sie selbst durch das Lesen des anderen Quelltexts gelernt? Was würden Sie jetzt bei Ihrem Quelltext anpassen?

* Wenn nötig mehr mit Funktionen arbeiten
* Zeile 20-23 ist redundant, da die E-Mail in 43ff validiert wird.
* Zeile 49-52 ist redundant, da Deutsch als Standard ausgewählt ist.
* Fehlermeldungen von Zeile 12-35 werden nicht ausgegeben da ein Pop-Up Fenster erscheint.

*Review des anderen Teams:*

Index

- Statusmeldung für Fehler bei NL-Anmeldung kommt, aber bei Erfolg sehe ich leider keine obwohl es im Code versucht wird. Scheinbar Bug

- Zugangsdaten für DB in Konstanten ablegen und diese dann in jedem Verbindungsaufbau nutzen. Erspart einem in größeren Projekten langes Durchsuchen des Codes da man dann nur an einer Stelle anpassen muss

- Verbindungsaufbau zu DB auszulagern kann clever sein, nur wird bei dem include eine query ausgeführt mit der nichts gemacht wird und direkt nach dem include nochmals link neu definiert über erneuten Verbindungsaufbau

- Dinge wie Berechnung der Counter kann man gut in Funktionen auslagern damit man diese wiederverwenden und leichter anpassen kann. Ist zudem übersichtlicher

NL Admin

- searchTerm wird nur definiert wenn GET Param gesetzt. Beim ersten und weiteren Aufrufen ohne GET Param sorgt dies für Probleme

- Die Auswahl der Sortierung sollte direkt dort stattfinden wo sie "applied" werden kann, zB über Button. Sonst muss man suchen

- Sortierung nach Name funktioniert nicht ganz. Hatte Name "test" drin und Zabian war davor. Evtl nicht Case insensitive?

Positives:

- Gute Aufteilung in seperate Files&Directories

- Sehr ansprechende visuelle Elemente wie Farbverläufe, Icons und abgerundete Ecken

- Clever umgesetzte dynamische Elemente wie einblendbare Allergenliste und Umrandung bei hover

- Nutzung von Serialize für Persistierung der Newsletterdaten

- Gute Nutzung von HTML Tags => Keine "Div-Suppe"

**Aufgabe 8**

*8.1*

**ALTER** **TABLE** gericht\_hat\_kategorie

**ADD** **CONSTRAINT** gericht\_hat\_kategorie **UNIQUE**(gericht\_id, kategorie\_id)

*8.2*

**ALTER** **TABLE** gericht

**ADD** **INDEX** idx\_name (**name**);

*8.3*

*1)*

**ALTER** **TABLE** gericht\_hat\_allergen

**ADD** **CONSTRAINT** delete\_from\_allergen

**FOREIGN** **KEY** (gericht\_id)

**REFERENCES** gericht (id)

**ON** **DELETE** **CASCADE;**

*2)*

**ALTER** **TABLE** gericht\_hat\_kategorie

**ADD** **CONSTRAINT** delete\_from\_kategorie

**FOREIGN** **KEY** (gericht\_id)

**REF**

**ERENCES** gericht (id)

**ON** **DELETE** **CASCADE;**

*8.4*

*1)*

**ALTER** **TABLE** gericht\_hat\_kategorie

**add** **CONSTRAINT** kategorie\_in\_verwendung **FOREIGN** **KEY**(Kategorie\_id) **REFERENCES** kategorie(id)

**ON** **DELETE** **RESTRICT**

*2)*

**ALTER** **TABLE** kategorie

**ADD** **CONSTRAINT** kategorie\_kategorie\_eltern\_id\_fk

**FOREIGN** **KEY** (eltern\_id) **REFERENCES** kategorie(id);

*8.5*

**ALTER** **TABLE** gericht\_hat\_allergen

**add** **CONSTRAINT** change\_allergen **FOREIGN** **KEY**(**code**) **REFERENCES** allergen(**code**)

**ON** **UPDATE** **CASCADE**;

*8.6*

**ALTER** **TABLE** gericht\_hat\_kategorie

**ADD** **PRIMARY** **KEY** (gericht\_id,Kategorie\_id)